**Problem1:**

For the first question, instead of using any statistical packages, I manually implemented the exponentially weighted covariance matrix following the steps from the in-class handouts. First, we can get the variance at time t is updated as:

$$\sigma_t^2 = \lambda\sigma_{t-1}^2 + (1 - \lambda)\left(x_{t-1} - \bar{x}\right)^2$$

Then sets the weights as:

$$w_{t-i} = (1 - \lambda)\,\lambda^{i-1}$$

, which are normalized to sum to 1 over the finite horizon as below:

$$\lim_{n \to \infty} (1 - \lambda) \sum_{i=1}^{n} \lambda^{i-1} = 1$$

$$\widehat{w_{t-i}} = \frac{w_{t-i}}{\sum_{j=1}^{n} w_{t-j}}$$

Then we can get the covariance between two assets x and y is calculated using the exponentially weighted deviations from their respective means:

$$\widehat{cov(x, y)} = \sum_{i=1}^{n} w_{t-i}\left(x_{t-i} - \bar{x}\right)\left(y_{t-i} - \bar{y}\right)$$

For this problem, we can choose to use 0.94,0.97 and 0.99 as selected lambdas. After applying those to the codes, we get the results like below:

Covariance Matrix for λ = 0.8:

[[ 4.77589227e-05  1.80777502e-05  4.79460277e-05 ...  5.68215072e-05

  5.97427094e-05  3.28392189e-05]

[ 1.80777502e-05  6.93452586e-05  3.39918871e-05 ...  5.51114416e-05

  1.76960182e-05 -1.19538922e-05]

[ 4.79460277e-05  3.39918871e-05  7.68612696e-05 ...  4.37213332e-05

  3.33875107e-05  6.22931380e-06]

...

[ 5.68215072e-05  5.51114416e-05  4.37213332e-05 ...  1.60530826e-04

  1.27496792e-04  5.17866206e-05]

[ 5.97427094e-05  1.76960182e-05  3.33875107e-05 ...  1.27496792e-04

  1.89320855e-04  1.66278038e-05]

[ 3.28392189e-05 -1.19538922e-05  6.22931380e-06 ...  5.17866206e-05

  1.66278038e-05  2.39850502e-04]]

Eigenvalues:

[8.95559335e-03 2.22535487e-03 1.73341659e-03 1.43667228e-03

9.94641983e-04 7.76857017e-04 4.90545956e-04 4.38399036e-04

4.00034573e-04 2.78307436e-04 2.48728141e-04 2.10519865e-04

1.93552566e-04 1.36331045e-04 1.24741932e-04 8.80947230e-05

7.40273695e-05 6.40624728e-05 5.13128827e-05 4.42194306e-05

4.13619644e-05 3.86613720e-05 3.25373611e-05 2.24750374e-05

1.92138848e-05 1.31393256e-05 1.09414399e-05 7.75603256e-06

5.17147532e-06 4.65103067e-06 2.62361899e-06 1.87736220e-06

1.55875038e-06 1.31949658e-06 8.78938014e-07 5.81067637e-07

3.64549851e-07 2.82462411e-07 2.42102076e-07 1.85582389e-07

1.67090298e-07 1.27347335e-07 9.20500722e-08 7.84190020e-08

5.88001803e-08 4.18361476e-08 3.73454782e-08 3.05630527e-08

2.47007835e-08 2.42078715e-08 1.38743045e-08 1.25854438e-08

9.39530541e-09 7.86402557e-09 6.50983221e-09 4.76856701e-09

3.75212587e-09 1.92995867e-09 1.78988269e-09 1.42484588e-09

1.26055668e-09 9.30838956e-10 5.63210011e-10 4.70461587e-10

3.83349687e-10 3.18087425e-10 2.26923629e-10 1.60705688e-10

1.35576712e-10 9.21469182e-11 8.20198699e-11 6.11741524e-11

4.12808360e-11 2.82061911e-11 2.50060627e-11 1.80908669e-11

1.58996597e-11 1.19170363e-11 9.23065010e-12 6.08543921e-12

5.68818781e-12 4.48086467e-12 3.99543651e-12 2.78097945e-12

2.49369510e-12 1.59297732e-12 1.05253911e-12 8.24019866e-13

4.53402979e-13 3.40387724e-13 1.59661341e-13 1.37500712e-13

1.07595692e-13 8.75724070e-14 6.46330460e-14 4.75331385e-14

2.32433657e-14 1.67823632e-14 3.36855746e-15 1.20988876e-15]

Eigenvectors:

[[ 0.07086368  0.0125327  -0.01411373 ... -0.08448692 -0.36349506

 -0.88422637]

[ 0.02137913 -0.07024505 -0.13160533 ... -0.01737612  0.03982206

  0.08420486]

[ 0.076117   -0.06061557 -0.0210185  ... -0.12408602  0.01558836

 -0.00663199]

...

[ 0.07562968  0.05604132 -0.09304905 ... -0.12250437 -0.00800378

  0.0300839 ]

[ 0.08109478  0.07429547  0.06224211 ... -0.07055294  0.02622302

  0.04681611]

[ 0.04488189  0.17386145 -0.07064461 ...  0.04105774  0.02592652

  0.01050223]]

Simulated Returns (first 5) for λ:

[[ 0.00219069  0.0052265   0.00817899  0.01040614 -0.01472081]

[ 0.00227819  0.00536858  0.00430607  0.0184067  -0.00462068]

[ 0.0040058  -0.00308079  0.01353142  0.01186227 -0.0202425 ]

[-0.00968682  0.00572739  0.02047359  0.02119107 -0.04275394]

[ 0.00860331  0.01674431  0.03942625  0.02110612 -0.07899666]]

Covariance Matrix for λ = 0.9:

[[7.83913946e-05 5.57219958e-05 7.34933108e-05 ... 1.04798201e-04

 8.33795892e-05 5.80715560e-05]

[5.57219958e-05 1.13615148e-04 6.56245673e-05 ... 7.99383778e-05

 4.38598941e-05 2.68095724e-05]

[7.34933108e-05 6.56245673e-05 1.10820578e-04 ... 8.54575815e-05

 5.18350640e-05 3.83475574e-05]

...

[1.04798201e-04 7.99383778e-05 8.54575815e-05 ... 2.98447262e-04

 1.78027756e-04 8.89882936e-05]

[8.33795892e-05 4.38598941e-05 5.18350640e-05 ... 1.78027756e-04

 2.27622783e-04 2.88720176e-05]

[5.80715560e-05 2.68095724e-05 3.83475574e-05 ... 8.89882936e-05

 2.88720176e-05 2.83084265e-04]]

Eigenvalues:

[1.23384383e-02 2.33995302e-03 1.89604082e-03 1.56558956e-03

 1.27509821e-03 1.17467694e-03 9.53830384e-04 8.64954709e-04

 6.34829782e-04 5.80633704e-04 5.25463981e-04 4.70897217e-04

4.34086408e-04 3.99496223e-04 3.40478125e-04 3.17176862e-04

2.82283907e-04 2.60786803e-04 2.52418454e-04 2.41717534e-04

1.75541075e-04 1.48215306e-04 1.37083176e-04 1.26957642e-04

1.15038103e-04 1.03304715e-04 9.48545522e-05 8.00632204e-05

7.90405007e-05 6.47275114e-05 5.37217316e-05 4.83177528e-05

4.29919443e-05 2.70759254e-05 2.64486523e-05 2.16966395e-05

1.80595611e-05 1.60866817e-05 1.34938309e-05 1.30046256e-05

1.16532923e-05 9.07665200e-06 8.26993063e-06 7.01391623e-06

6.52003483e-06 5.85246298e-06 5.08642372e-06 4.07315153e-06

3.37679419e-06 3.22975039e-06 3.01390566e-06 2.88147513e-06

2.41300130e-06 2.20522793e-06 2.03397926e-06 1.42628750e-06

1.29844095e-06 1.17792080e-06 1.09939105e-06 9.31451451e-07

7.60131299e-07 6.67784172e-07 5.37170669e-07 4.86825147e-07

4.37122433e-07 3.67897464e-07 3.15330456e-07 2.54593668e-07

2.46840723e-07 2.46081493e-07 2.10029658e-07 1.68896539e-07

1.42615138e-07 1.38491144e-07 1.08309571e-07 9.31669604e-08

7.67101656e-08 6.61480362e-08 6.31058725e-08 5.11615726e-08

3.80755365e-08 3.61891363e-08 3.11528710e-08 2.76531725e-08

1.98006483e-08 1.79979279e-08 1.43894135e-08 9.27296753e-09

8.60554952e-09 5.57031160e-09 5.43192468e-09 4.74417170e-09

4.29014199e-09 3.85935611e-09 3.26371231e-09 2.71267502e-09

1.91828083e-09 1.22085155e-09 1.89152642e-10 9.28046401e-11]

Eigenvectors:

[[ 0.07664865  0.00204825  0.02135515 ...  0.06548911  0.20848143

  0.94477831]

[ 0.04661484  0.01553862  0.08144472 ... -0.03264211 -0.02859407

-0.0867698 ]

[ 0.07146968 -0.01070067  0.02821677 ... -0.09085497 -0.02332946

0.01263465]

...

[ 0.1047264  -0.07391598  0.06706017 ... -0.05361415 -0.00278918

0.0070176 ]

[ 0.08130214 -0.08979608  0.07803657 ... -0.04780004 -0.01081958

-0.03182632]

[ 0.05394819  0.05758493 -0.02516378 ...  0.07995014 -0.01313524

-0.02650987]]

Simulated Returns (first 5) for λ:

[[ 0.01625309  0.00462238 -0.0039461   0.00672467 -0.00786345]

[ 0.01275624 -0.00152733 -0.00933995 -0.00444239  0.00730991]

[ 0.01605414  0.00192634 -0.00399384 -0.00421832 -0.00636205]

[ 0.01569573  0.02356535 -0.00404289  0.03579492 -0.01534951]

[ 0.02361054  0.02445811 -0.0388742   0.00495262 -0.04590711]]

Covariance Matrix for λ = 0.94:

[[9.24874550e-05 7.67064710e-05 8.70550156e-05 ... 1.21037304e-04

8.82093995e-05 6.73769433e-05]

[7.67064710e-05 1.60703693e-04 8.82501866e-05 ... 8.29726483e-05

4.91381471e-05 4.74368537e-05]

[8.70550156e-05 8.82501866e-05 1.35036417e-04 ... 9.95235708e-05

5.61916131e-05 5.44816044e-05]

...

[1.21037304e-04 8.29726483e-05 9.95235708e-05 ... 3.82144131e-04

1.87715924e-04 1.00198479e-04]

[8.82093995e-05 4.91381471e-05 5.61916131e-05 ... 1.87715924e-04
2.40607733e-04 3.92272032e-05]
[6.73769433e-05 4.74368537e-05 5.44816044e-05 ... 1.00198479e-04
3.92272032e-05 2.57721419e-04]]
Eigenvalues:
[1.37801875e-02 3.01448756e-03 2.27379621e-03 1.41386621e-03
1.40088095e-03 1.23554489e-03 1.08090654e-03 1.03926412e-03
9.21646019e-04 7.64783086e-04 6.68827853e-04 5.53066122e-04
5.11771624e-04 4.89413160e-04 4.24688664e-04 4.04335304e-04
3.61264162e-04 3.55552884e-04 3.16984351e-04 2.96120002e-04
2.69107584e-04 2.26264674e-04 2.10676343e-04 1.98879012e-04
1.90319309e-04 1.64483636e-04 1.55041488e-04 1.53056493e-04
1.39153743e-04 1.22520326e-04 1.16203536e-04 1.09600691e-04
8.98558792e-05 8.12926787e-05 7.52614793e-05 6.77175295e-05
5.80350495e-05 5.33882107e-05 4.92012259e-05 4.35638515e-05
4.20592593e-05 3.86501098e-05 3.51689379e-05 3.40001814e-05
2.79528483e-05 2.71551473e-05 2.36777878e-05 2.26486732e-05
2.07156381e-05 1.95270560e-05 1.70992699e-05 1.55729838e-05
1.41108556e-05 1.32886823e-05 1.28765969e-05 1.11225427e-05
1.02440576e-05 9.18269164e-06 8.70723200e-06 7.69226080e-06
6.63336302e-06 6.46713074e-06 6.14375325e-06 5.13331487e-06
4.76850230e-06 4.63188577e-06 4.12641755e-06 3.81162776e-06
3.51831249e-06 3.24824426e-06 2.80028502e-06 2.34616652e-06
2.25277079e-06 2.17675044e-06 2.06062787e-06 1.41187440e-06
1.29658129e-06 1.19410169e-06 1.13814966e-06 9.67281195e-07
8.73127416e-07 7.83695521e-07 7.34912581e-07 6.67472463e-07

6.11642341e-07 4.82093201e-07 4.15929937e-07 3.56066513e-07

3.04495693e-07 2.50108702e-07 2.23466721e-07 2.03976331e-07

1.96236138e-07 1.84171015e-07 1.57664135e-07 1.39256017e-07

9.71444743e-08 7.58133500e-08 9.51661257e-09 4.55513276e-09]

Eigenvectors:

[[-0.07835271 -0.01253075 -0.02119607 ... 0.06037998 0.27593961

0.93608353]

[-0.05484338 0.0024911 -0.06107574 ... -0.03302829 -0.03256161

-0.08155373]

[-0.07079584 -0.01283305 -0.02394726 ... -0.0450367 -0.02782043

-0.00375403]

...

[-0.1086942 -0.11902557 0.02207318 ... -0.03076291 -0.01121685

0.01091622]

[-0.07688112 -0.11973829 0.01699196 ... -0.05148978 -0.00415859

-0.02110419]

[-0.05290291 -0.01905144 -0.0335888 ... 0.06813575 -0.0085687

-0.03063525]]

Simulated Returns (first 5) for λ:

[[ 0.00099202 -0.01136929 0.0055886 -0.00739003 -0.00612879]

[ 0.00796354 -0.00177834 0.00867099 -0.00046039 -0.01291602]

[-0.00027812 -0.01344849 0.01096402 -0.00067018 -0.0039686 ]

[-0.020735 -0.02966279 0.01531684 -0.0167787 -0.00499268]

[-0.01169055 -0.04588317 0.00637121 0.00988191 -0.02058596]]

Covariance Matrix for λ = 0.97:

[[8.77475071e-05 8.21306434e-05 8.71286167e-05 ... 1.07263825e-04

7.71446745e-05 6.12271416e-05]

[8.21306434e-05 2.12476503e-04 1.00647212e-04 ... 6.76861508e-05

4.16016033e-05 5.23388083e-05]

[8.71286167e-05 1.00647212e-04 1.50749075e-04 ... 8.47124558e-05

4.90324563e-05 5.52897389e-05]

...

[1.07263825e-04 6.76861508e-05 8.47124558e-05 ... 3.98294179e-04

1.73177872e-04 8.85383355e-05]

[7.71446745e-05 4.16016033e-05 4.90324563e-05 ... 1.73177872e-04

2.47930308e-04 3.92215596e-05]

[6.12271416e-05 5.23388083e-05 5.52897389e-05 ... 8.85383355e-05

3.92215596e-05 2.03520609e-04]]

Eigenvalues:

[1.23941489e-02 3.64500172e-03 2.04002145e-03 1.55488659e-03

1.33860836e-03 1.23826994e-03 1.05130810e-03 9.46478204e-04

8.65463314e-04 7.97445974e-04 6.83858715e-04 6.11823340e-04

5.58623277e-04 5.48715949e-04 5.06585935e-04 4.76337314e-04

4.33384598e-04 4.00018519e-04 3.54116145e-04 3.43973395e-04

3.22046555e-04 3.04503159e-04 2.76080599e-04 2.63460621e-04

2.32913611e-04 2.19134737e-04 2.10758712e-04 1.96597684e-04

1.84976369e-04 1.79407851e-04 1.62489477e-04 1.56570548e-04

1.48114342e-04 1.41957098e-04 1.29217313e-04 1.24310714e-04

1.19474763e-04 1.13043939e-04 1.10392012e-04 1.01845444e-04

9.44529174e-05 8.62544818e-05 8.18964264e-05 7.93580508e-05

7.30472318e-05 6.78643350e-05 6.58940620e-05 6.15155038e-05

5.47881291e-05 5.25537157e-05 4.91700970e-05 4.77064057e-05

4.46839812e-05 4.27090463e-05 3.86120823e-05 3.70192668e-05

3.58930596e-05 3.32146937e-05 3.18698020e-05 3.06916586e-05

2.77262261e-05 2.74965084e-05 2.58716291e-05 2.52498490e-05

2.25772818e-05 2.24603836e-05 1.98469593e-05 1.91544993e-05

1.73193255e-05 1.63607747e-05 1.55504336e-05 1.36767550e-05

1.29610901e-05 1.20919007e-05 1.15287733e-05 1.02571429e-05

9.37914056e-06 8.76110472e-06 8.32352983e-06 7.90247407e-06

7.13090130e-06 6.73103263e-06 6.50207420e-06 5.93113271e-06

5.48112908e-06 5.27325837e-06 4.70129413e-06 3.92956630e-06

3.70832102e-06 3.24283592e-06 3.13235746e-06 3.02416074e-06

2.74294036e-06 2.47002664e-06 2.31552232e-06 2.04083254e-06

1.57688103e-06 1.42413324e-06 1.21050057e-07 6.17188013e-08]

Eigenvectors:

[[ 7.99267545e-02 -1.60027528e-02  2.44606437e-02 ...  2.71155968e-02

 -1.91598731e-01  9.64764374e-01]

[ 6.29529917e-02  1.04975096e-02  6.70548364e-02 ...  2.35695147e-02

  1.87339756e-02 -7.62961332e-02]

[ 7.44344538e-02 -7.22584353e-04  2.76527992e-02 ...  3.22179813e-02

  2.03151703e-02 -3.05078957e-02]

...

[ 1.05370009e-01 -1.26446175e-01 -4.40906219e-02 ...  9.64971040e-03

  7.71249407e-03  4.87141582e-03]

[ 7.26843979e-02 -1.22177554e-01 -6.46467092e-02 ... -5.86844064e-02

  7.86505270e-03 -1.03611790e-02]

[ 5.06510223e-02 -3.86658754e-02  4.22290253e-02 ...  4.55271615e-02

  5.48822258e-03 -3.12938280e-02]]

Simulated Returns (first 5) for λ:

[[ 0.00278182 -0.00973356  0.00142448 -0.00478148  0.00083275]

[ 0.01407607 -0.00485422  0.01310555 -0.01869864  0.02314863]

[ 0.00622869 -0.02053096 -0.00039556 -0.00157567 -0.00995009]

[ 0.01861959  0.00238141 -0.00751578 -0.02427128 -0.00496752]

[ 0.00877019 -0.02173858  0.00768469 -0.0891445  -0.0131204 ]]

Covariance Matrix for λ = 0.99:

[[6.92988938e-05 6.98547004e-05 7.52824019e-05 ... 7.48683265e-05

 6.51707063e-05 4.57603414e-05]

[6.98547004e-05 2.23728345e-04 9.20840581e-05 ... 3.93872409e-05

 3.26746470e-05 4.22524666e-05]

[7.52824019e-05 9.20840581e-05 1.54733113e-04 ... 4.73537887e-05

 3.60261515e-05 4.52209662e-05]

...

[7.48683265e-05 3.93872409e-05 4.73537887e-05 ... 3.64324230e-04

 1.61415691e-04 6.12198437e-05]

[6.51707063e-05 3.26746470e-05 3.60261515e-05 ... 1.61415691e-04

 2.74838925e-04 3.56132907e-05]

[4.57603414e-05 4.22524666e-05 4.52209662e-05 ... 6.12198437e-05

 3.56132907e-05 1.46196433e-04]]

Eigenvalues:

[8.83076474e-03 3.45511519e-03 1.52008745e-03 1.31932451e-03

 1.07554100e-03 1.00728001e-03 8.81962758e-04 7.77640635e-04

 6.99457176e-04 6.75009304e-04 6.06011724e-04 5.43217727e-04

 5.22608635e-04 4.77322347e-04 4.53979583e-04 4.42106884e-04

 3.97471395e-04 3.96166703e-04 3.81480479e-04 3.66313217e-04

3.30650584e-04 3.17204804e-04 3.10965244e-04 2.98603558e-04

2.73911806e-04 2.71892570e-04 2.59570461e-04 2.47858804e-04

2.26387266e-04 2.24386604e-04 2.07405441e-04 2.00971706e-04

1.98581545e-04 1.83540896e-04 1.76122792e-04 1.70233375e-04

1.62225405e-04 1.58280286e-04 1.51328531e-04 1.47266804e-04

1.41768782e-04 1.30493429e-04 1.27112747e-04 1.22010665e-04

1.14527217e-04 1.08363252e-04 1.05802828e-04 1.03335913e-04

1.01542130e-04 9.88650691e-05 9.07045926e-05 8.49487831e-05

7.97149356e-05 7.87231158e-05 7.54832623e-05 7.32600518e-05

7.20554416e-05 6.83842253e-05 6.64648636e-05 5.96405777e-05

5.76391630e-05 5.50453780e-05 5.40249846e-05 5.28623258e-05

5.15260970e-05 4.94604533e-05 4.90977996e-05 4.29687437e-05

4.04261297e-05 3.91709832e-05 3.78014909e-05 3.58197476e-05

3.39276300e-05 3.29888463e-05 3.08349754e-05 3.02499881e-05

2.88873255e-05 2.71090169e-05 2.59125668e-05 2.49566322e-05

2.25299643e-05 2.14579360e-05 1.99214985e-05 1.95284767e-05

1.85685990e-05 1.79505998e-05 1.66510045e-05 1.59557550e-05

1.52688484e-05 1.40693599e-05 1.21466604e-05 1.16408148e-05

1.03202702e-05 9.47077011e-06 8.69590607e-06 7.41349299e-06

6.97608735e-06 6.59049572e-06 3.57897239e-07 2.47337046e-07]

Eigenvectors:

[[-8.37492355e-02  1.67426720e-02  4.25224459e-02 ...  1.32961905e-02

  1.89359870e-01 -9.69013509e-01]

[-7.19293053e-02 -1.11530472e-02  6.69367419e-02 ... -3.45691529e-02

 -1.59680042e-02  6.49073503e-02]

[-8.45558055e-02 -1.95563527e-02  1.15549375e-01 ...  9.04950982e-02

-1.05710891e-02  4.87532768e-02]

...

[-1.00119492e-01  1.42837667e-01 -1.37375862e-01 ...  8.15796978e-03

  5.03064958e-04 -4.95820506e-03]

[-8.22916798e-02  1.49249588e-01 -7.74295726e-02 ...  6.70301958e-02

 -6.67176120e-03  6.68854395e-03]

[-4.89763653e-02  4.24901964e-02  4.52175052e-02 ...  5.56067853e-02

 -3.33373084e-03  2.49730400e-02]]

Simulated Returns (first 5) for λ:

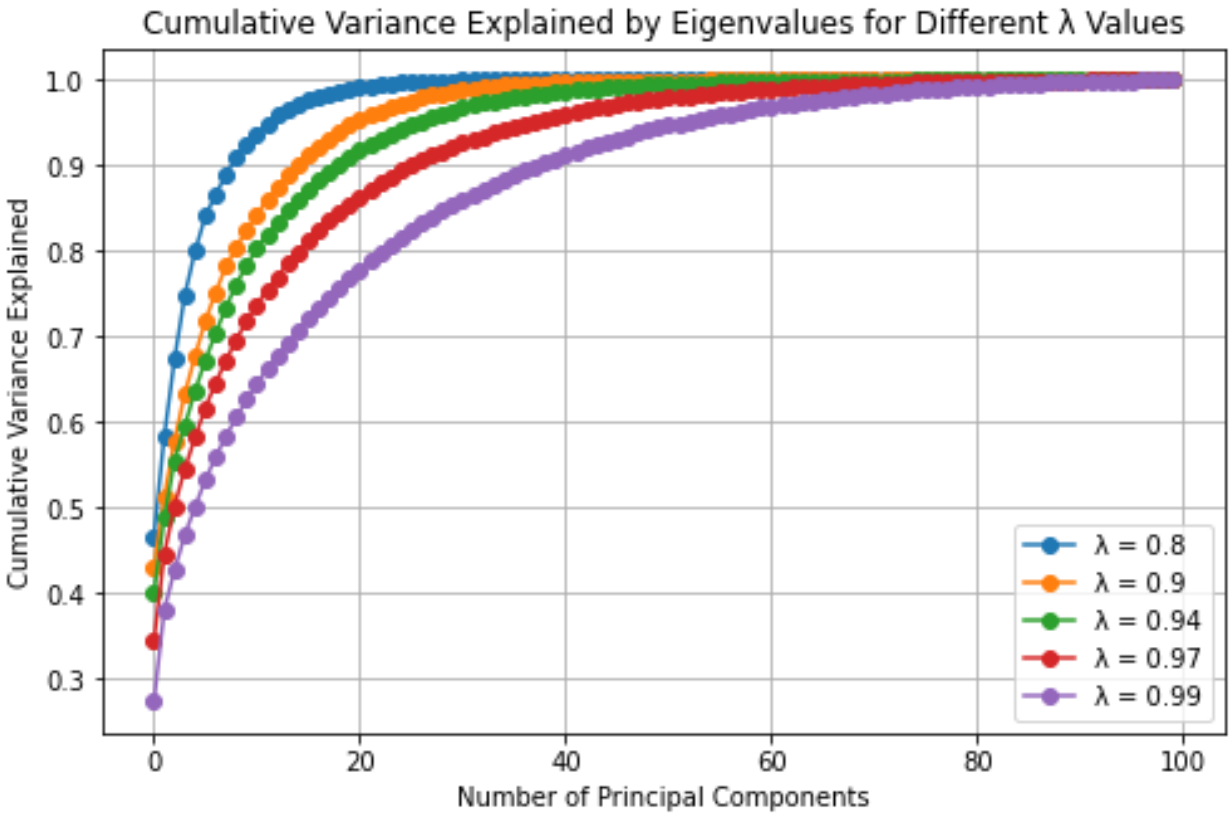
[[-0.00057748  0.00832601  0.00058951  0.0014156   0.00446018]

[-0.00686546  0.01384919  0.02991339 -0.00121995 -0.00294819]

[ 0.00301891  0.00768929  0.00719375  0.010528    0.00719259]

[-0.01046685  0.0009455  -0.01021482 -0.003914   -0.00261355]

[ 0.03387756  0.07723555  0.01598682  0.01588962  0.02254696]]




Based on the three charts, we can see that smaller lambda values provide more weights to recent returns, which makes the result that the covariance matrix reflects recent fluctuations more than long-term trends. As lambda increases, the covariance matrix becomes more stable and less sensitive to recent changes, the eigenvalues are more evenly distributed, where we need to consider more elements into the total variance. Here we can use lower lambda value cases to capture the most recent or rapid market changes but might overfit the short-term part; while we can also use the more stable larger lambda values for long-term market analysis but might miss the short-term changes.

## Problem 2:

This question I implemented chol_psd(), and near_psd() functions in python based on the codes in the repository. In the chol_psd() function, we did Cholesky decomposition assuming that the matrix is positive definite, where we breakdown the matrix into product of lower triangular matrix and its transpose. For the near_psd() function we performed the Cholesky decomposition for positive semi-definite and deal with the two reasons for non-PSD, including the floating point error and missing data. We used np.linalg.eigvals to check if any eigenvalues are negative, if so then the matrix is not PSD. In the near_psd() function,

we implemented the Higham's method to find the nearest positive semi-definite matrix, here we set any negative eigenvalues to zero to make sue we have PSD matrix. We then used PCA to do the simulation for random variables only using the positive eigenvalues. Both generate_with_missing and missing_cov handle the missing values. We also compare the results of both using the Frobenius Norm and compare the run time between the two.

***For the runtime, we get the following outputs:***

| | n | Near PSD Time (s) | Higham PSD Time (s) | Frobenius Norm (Near PSD) \ |
|---|---|---|---|---|
| 0 | 500 | 0.028993 | 3.001006 | 0.627523 |
| 1 | 800 | 0.107001 | 9.764999 | 0.795801 |
| 2 | 1000 | 0.195997 | 18.546707 | 0.890491 |

| | Frobenius Norm (Higham PSD) |
|---|---|
| 0 | 0.063643 |
| 1 | 0.063889 |
| 2 | 0.063971 |

As you can see from the results, Near PSD can take much less runtime for no matter small or large size of matrices compared to Higham PSD, and as the size increases, the difference between the runtime becomes larger, meaning that from the runtime's perspective, Near PSD is better.

Let's take a look at the Frobenius Norms between Near PSD and Higham PSD. I ran different test cases to see what happened when we have increasing n. As we can see from the table, for each same n, the Frobenius norms for Higham PSD is always less than Near PSD, meaning that the Higham method can provide a more accurate simulation. And as we have increasing n, the Frobenius norms for both Near PSD and Higham PSD keep increasing, meaning that as we have larger matrices, the accuracy level for both methods is lowering; however, Higham PSD has smaller changes than the Near PSD for the same level of n increasing, meaning that we always get higher accuracy for using Higham PSD.

***Near PSD:***

- Pros:
    - Significantly faster, especially dealing with large size of matrices.
    - The computation is easier so good to use for large-scale problems.

- Cons:
    - This method might not consider that much as the Higham method.

***Higham PSD:***

- Pros:
    - The method is better to handle extreme edge problems
    - The method can provide more accuracy simulation.
- Cons:
    - The speed is lower, especially dealing with large-scale problems
    - The method might cost more memory usage when trying to run it because of the iterative approach.

## Problem 3:

First we used the in-class handouts materials to generate Pearson and Exponentially Weighted Correlation Matrices, then we ran the PCA simulation from the Cholesky decomposition and PCA-based simulation. After that, we calculated the Frobenius Norm, where we got sum of squared differences between the original covariance matrix and the simulated covariance matrix, lower number means higher accuracy between the simulated and original one.

Then we generated results like below:

| | Matrix Type | Method | Time (s) | Frobenius Norm |
|---|---|---|---|---|
| 0 | Pearson Corr + Standard Var | Direct | 0.136001 | 8.635591e-08 |
| 1 | Pearson Corr + Standard Var | PCA 100% | 0.095000 | 4.507415e-06 |
| 2 | Pearson Corr + Standard Var | PCA 75% | 0.024002 | 4.507415e-06 |
| 3 | Pearson Corr + Standard Var | PCA 50% | 0.017986 | 4.507415e-06 |
| 4 | Pearson Corr + EW Var | Direct | 0.101001 | 1.692487e-07 |
| 5 | Pearson Corr + EW Var | PCA 100% | 0.090999 | 1.132394e-05 |
| 6 | Pearson Corr + EW Var | PCA 75% | 0.017000 | 1.132394e-05 |
| 7 | Pearson Corr + EW Var | PCA 50% | 0.019995 | 1.132394e-05 |
| 8 | EW Corr + Standard Var | Direct | 0.106995 | 8.823302e-11 |
| 9 | EW Corr + Standard Var | PCA 100% | 0.099992 | 6.365738e-09 |
| 10 | EW Corr + Standard Var | PCA 75% | 0.020003 | 6.365738e-09 |

11 EW Corr + Standard Var PCA 50% 0.024013 6.365738e-09

12 EW Corr + EW Var Direct 0.114999 2.239712e-04

13 EW Corr + EW Var PCA 100% 0.099999 1.354789e-02

14 EW Corr + EW Var PCA 75% 0.018985 1.354791e-02

15 EW Corr + EW Var PCA 50% 0.021994 1.354791e-02

As we can see from the outputs, the direct Cholesky decomposition method has lower Frobenius norms, indicating better simulation than the PCA-based simulations. For the PCA-based simulations, as we have more principal components, we can get lower Frobenius norms.

When comparing the run times of the simulations, we find that direct simulation using Direct is generally slower than PCA-based simulations, particularly as the number of retained components decreases. On the other hand, PCA-based simulations, especially those with reduced components are significantly faster.

To summarize, we need to choose Direct if we have high priority for accuracy, and its more suitable for smaller-scale problems. We need to consider PCA-based simulations if we want to reach high efficiency for running the computation, and its preferable for large-scale problems.